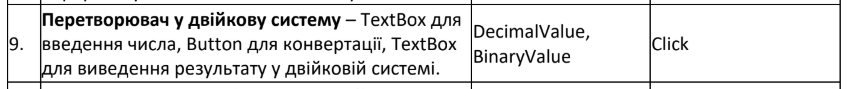
**Практична робота №1**

**РОЗРОБКА КОМПОНЕНТІВ**

**Мета роботи:** Вивчення особливостей створення та використання компонентів (components) та елементів інтерфейсу користувача (UI elements). Закріплення навичок об'єктно-орієнтованого програмування (OOP) з дотриманням принципів SOLID.

**Завдання**

****

Створимо нове рішення, додаємо додаток та бібліотеку(назвемо BinaryConverterControl).

**BinaryConverter.xaml:**

<UserControl x:Class="BinaryConverterControl.BinaryConverter"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Height="100" Width="250">

<Grid Margin="5">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal" Grid.Row="0" Margin="0,0,0,5">

<TextBox x:Name="InputTextBox" Width="100" Margin="0,0,5,0"/>

<Button Content="Convert" Click="ConvertButton\_Click"/>

</StackPanel>

<TextBox x:Name="ResultTextBox" Grid.Row="1" IsReadOnly="True" Width="200"/>

</Grid>

</UserControl>

**BinaryConverter.xaml.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace BinaryConverterControl

{

/// <summary>

/// Логика взаимодействия для BinaryConverter.xaml

/// </summary>

public partial class BinaryConverter : UserControl

{

public event EventHandler<string> BinaryValueClick;

public int DecimalValue

{

get => int.TryParse(InputTextBox.Text, out int value) ? value : 0;

set => InputTextBox.Text = value.ToString();

}

public BinaryConverter()

{

InitializeComponent();

}

private void ConvertButton\_Click(object sender, RoutedEventArgs e)

{

if (int.TryParse(InputTextBox.Text, out int value))

{

string binary = Convert.ToString(value, 2);

ResultTextBox.Text = binary;

BinaryValueClick?.Invoke(this, binary);

}

else

{

MessageBox.Show("Invalid number.");

}

}

}

}

**MainWindow.xaml:**

<Window x:Class="Task01.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:controls="clr-namespace:BinaryConverterControl;assembly=BinaryConverterControl"

Title="Binary Converter App" Height="150" Width="280">

<Grid Margin="10">

<controls:BinaryConverter x:Name="BinaryControl" BinaryValueClick="BinaryControl\_BinaryValueClick"/>

</Grid>

</Window>

**MainWindow.xaml.cs:**

using System.Text;

using System.Windows;

namespace Task01;

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

[System.Diagnostics.CodeAnalysis.SuppressMessage("Design", "CA1515:Unlike a class library, an application's API isn't typically referenced publicly, so types can be marked internal", Justification = "Required for XAML binding or external project references.")]

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void BinaryControl\_BinaryValueClick(object sender, string binaryValue)

{

MessageBox.Show($"Binary: {binaryValue}", "Conversion Result");

}

}

Як можемо побачити все працює коректно

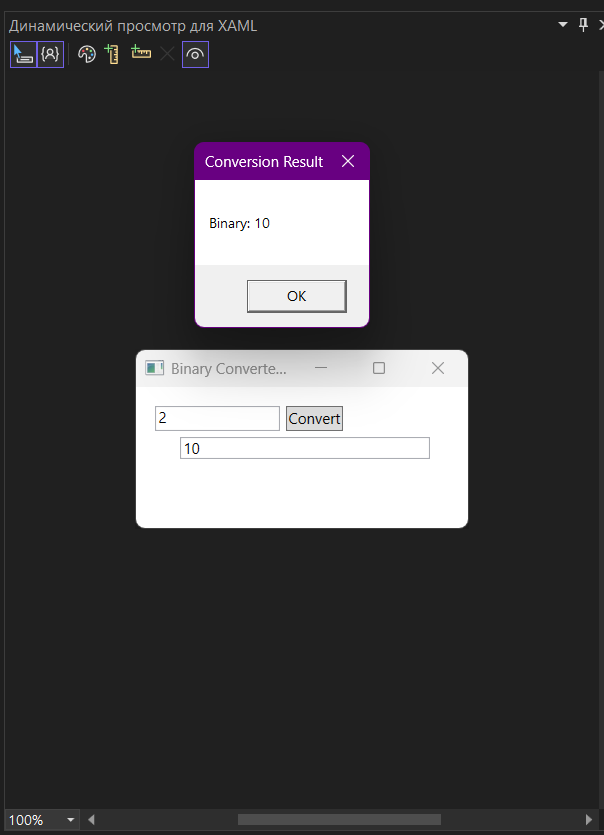


Рисунок 1.1 – Результат

**Висновок:** Вивчив особливості створення та використання компонентів (components) та елементів інтерфейсу користувача (UI elements). Закріпив навички об'єктно-орієнтованого програмування (OOP) з дотриманням принципів SOLID.